An Expert System for Space Shuttle Flight Control

July 9th, 1986

Fred Highland, IBM FSD
18100 Frederick Pike
Gaithersburg, MD 20879

Oscar W. Olszewski, NASA JSC
Houston, TX 77058

INTRODUCTION.

The United States Congress has become convinced that the National Aeronautics and Space Administration's (NASA) Space Station Program could be a logical user and a highly visible stimulator of new technological advances for the U.S. economy. To that end, Public Law 98-371 has mandated that 10% of the total Space Station costs be spent on advanced automation and artificial intelligence technologies (NASA, 1985). To obtain more experience to fulfill this congressional mandate, NASA has become active in automation endeavors in its Space Transportation System (STS) program which uses the Space Shuttle.

The process of monitoring Shuttle activities from the ground to ensure the success of the mission and the safety of the crew is called flight control. Shuttle flight control requires a complex configuration of computer and communications systems, and a large team of highly trained controllers to perform the mission monitoring and control function. The workload of these flight control teams is increasing with ~~the projected rate of Shuttle missions and~~ more complex payloads and At the same time the skills of flight controllers are in constant demand for new programs such as the Space Station. The large number of flight controllers required also drives up the cost of Shuttle operations.

One of the most promising solution approaches to the problems of Shuttle flight control is the use of expert systems technology. Expert systems, in theory, provide a means to capture the knowledge of experienced flight

controllers in a computer program. Unlike human controllers, an expert system will not fatigue under the stress of real-time monitoring, be transferred to another assignment, or retire. Unlike conventional computer programs, expert systems, with their separate knowledge bases, are easy to change as new knowledge and experience is acquired. However, expert systems lack common sense (Waterman et al, 1985) and therefore it is doubtful that they will totally replace their human counterparts. Use of this technology in an advisory role and as training aids would be more realistic.

While demonstration expert systems have shown potential in many application areas, the practical use of this technology in the Shuttle flight control environment poses some challenging problems. The real-time nature of the environment requires expert decisions to be made in a matter of seconds using hundreds of numerical and status parameters. These systems must also be embedded in the (MCC) software environment in order to obtain information efficiently from the surrounding system. The commercially available expert system tools are implemented in LISP or other (AI) languages and operate only on specially built computers. This prevents them from being embedded into a conventional hardware and software environment. They also have not demonstrated the performance necessary for real-time operation with large amounts of data.

The application of expert system technologies to the Shuttle flight control problem has been the subject of research conducted as part of FSD Houston's Spacecraft Control Centers Independent Research and Development project. This work has produced a prototype expert system tool, called FLIGHT CONTROLLER, which is designed to support the development of real-time expert systems applications embedded in and integrated with the future Shuttle MCC

environment. During 1984 and 1985 the prototype software was developed and the knowledge engineering of an entry navigation application begun. This lead to demonstrations of the prototype system to NASA and the USAF in 1985 and 1986. NASA is currently funding further research and development on the FLIGHT CONTROLLER system aimed at improving its level of expertise in the entry navigation application and rehosting the system to workstations already in the MCC.

In this article, we will discuss the FLIGHT CONTROLLER system, its design, application development, and status. We will first describe the entry navigation flight control task which the expert system performs. We will then describe the design of the FLIGHT CONTROLLER expert system software which was used for prototype development, the development approach used to create the knowledge base, and the user interface for the candidate application. Finally, we will discuss the results obtained from the prototype and the conclusions that we have drawn from this research.

THE SHUTTLE FLIGHT CONTROL APPLICATION.

Shuttle flight control is the process of monitoring the status and performance of the Shuttle's trajectory and onboard systems and taking the actions necessary to maintain the health of those systems. This task is accomplished by monitoring trajectory tracking data from various radar stations and the data from the Shuttle's telemetry downlink. All of this information is presented to controllers on video displays in real time. When problems are detected, corrective measures are determined and actions taken either by voice communication to the Shuttle crew or through changes to the Shuttle onboard systems through command uplink.

The Flight Director, who is in charge of the Shuttle mission, has a team of highly specialized personnel to assist him in decision making. During the landing phase, or entry, the Shuttle is basically a glider with no capability for last minute wave-off or turn-around. To assist the Flight Director in this very critical phase, the Guidance Officer provides advice in matters pertaining to Shuttle guidance and navigation. The Guidance Officer is himself advised on Shuttle navigation matters by the controllers at the Onboard Navigation (ONAV) console. The ONAV console landing phase monitoring for the primary navigation system was selected by this project as a prototype application domain because of the quality of the documentation, the level of expertise required, and because it best demonstrates the real-time capabilities of the system.

(ONAV)

The basic responsibilities of the Onboard Navigation console are to monitor the status of Shuttle navigation systems and ensure that the Shuttle's onboard computers always have the most accurate position and velocity information available. This is done by comparing the navigation information present in the Shuttle onboard computers (the "state") with information derived independently by ground radar sources. When problems are detected, they are isolated to the Shuttle system causing the error and the crew is advised to disconnect that component. If, because of time or other reasons, the problem cannot be isolated to a particular Shuttle system, the onboard position and velocity information is updated via a command uplink with the aid of the current ground computed navigation data.

The ONAV task requires the simultaneous monitoring of several components that contribute to the navigation state as illustrated in Figure 1. These include: Inertial Measurement Units (IMUs) that provide attitude and acceleration information, Tactical Air Navigation (TACAN) systems that provide range and bearing from a fixed point on the earth, the Air Data Transducer Assembly (ADTA) that provides barometric altitude, and the Microwave Landing System (MSBLS) for the final approach to the runway. In addition, the ground stations assessment as to the actual Shuttle position and velocity must also be monitored to determine if the Shuttle's navigation is sufficiently accurate for a safe landing. The monitoring process is further complicated by the presence of redundant instruments of all the components as well as two separate navigation systems (primary and backup). During a Shuttle entry, the ONAV flight controllers must scan over 150 data items on specialized CRT displays every 4 seconds for each of the two navigation systems, stay in voice contact with other members of the flight control team to determine the status of ground radar and other systems,

adjust for noise in the data, and make the necessary recommendations. The workload requirements of this task dictate that this job be done by 2 controllers - one to monitor the primary flight control system and one to monitor the backup system.

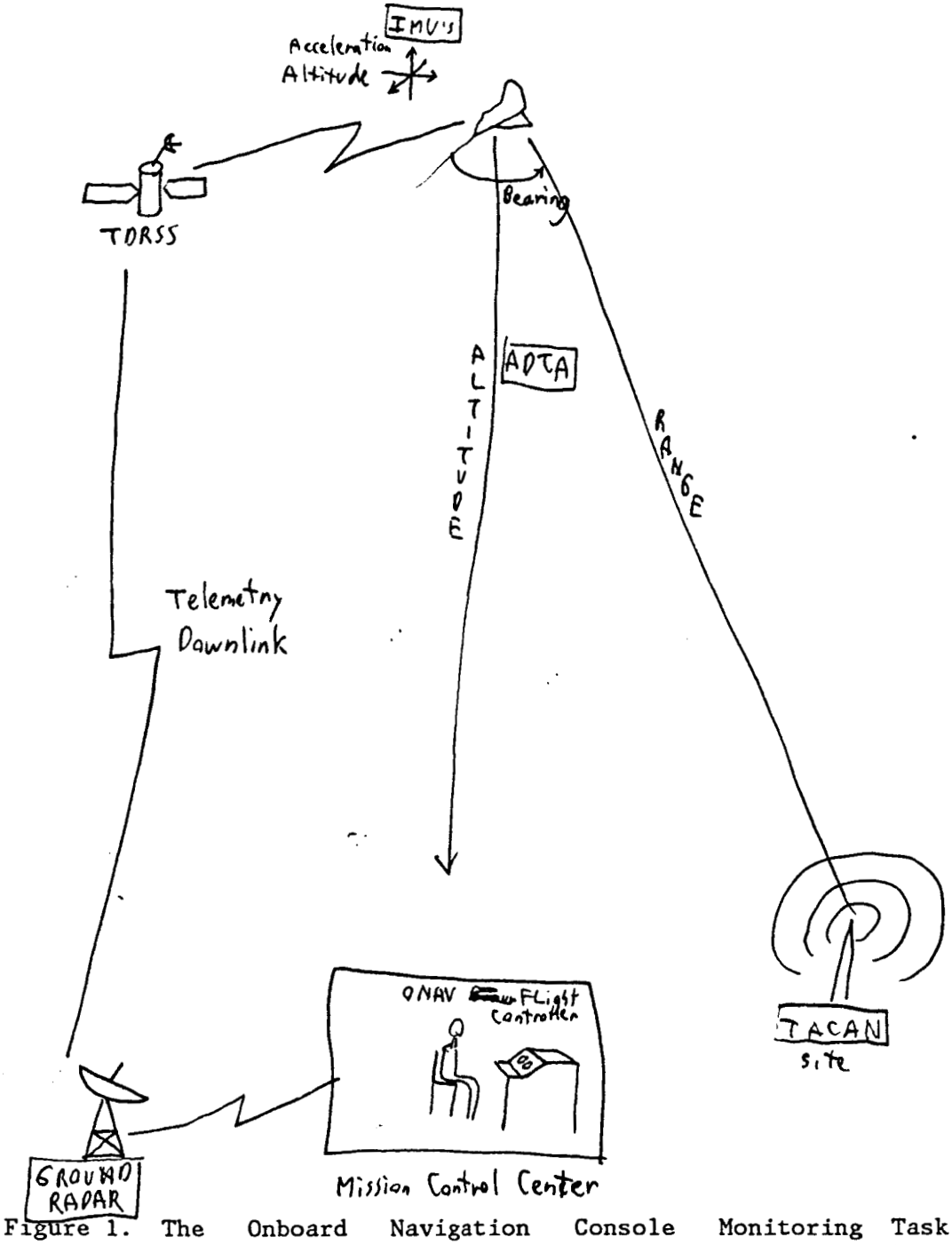


Figure 1. The Onboard Navigation Console Monitoring Task: The components that provide navigation information to the Shuttle are monitored in real time by Onboard Navigation console personnel to maintain a correct navigation position and advise the crew on use of the system.

THE FLIGHT CONTROLLER EXPERT SYSTEM.

The FLIGHT CONTROLLER system is a generalized tool that can be used to create real-time expert system applications integrated with the Shuttle mission environment. It is designed to operate as an embedded application in a general purpose flight control workstation like that proposed for NASA's MCC Upgrade program. The expert system requests and receives Shuttle data from the flight control host system via a Local Area Network (LAN), performs inferencing on the data, and presents recommendations on one of the workstations high resolution color displays. The prototype system was developed on a MASSCOMP MC500 workstation under the UNIX* operating system.

The FLIGHT CONTROLLER system is composed of two major components: the Inference System which performs real-time reasoning and system interfacing, and the Knowledge Acquisition System which supports the off-line definition and processing of the knowledge base for real-time use. A block diagram of major FLIGHT CONTROLLER components is shown in Figure 2. The details of each of these components are discussed in the following sections.

---

* UNIX is a trademark of Bell Laboratories, Inc.

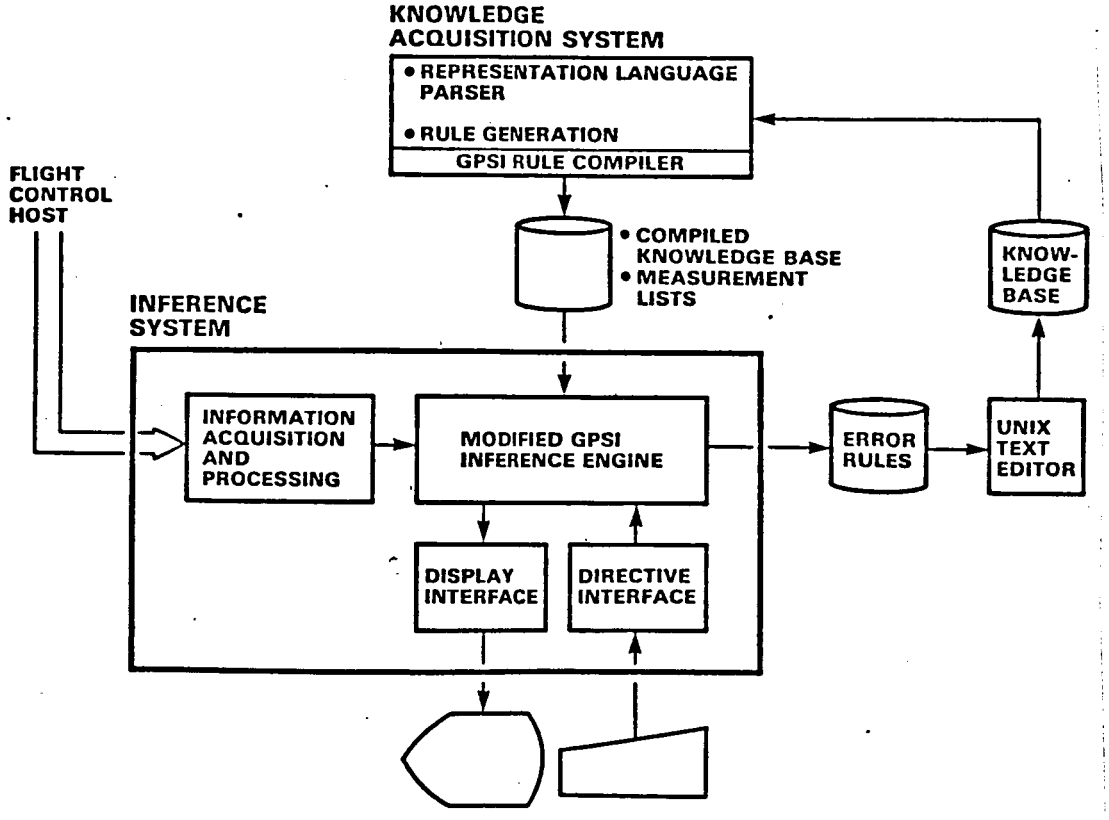


Figure 2. FLIGHT CONTROLLER System Architecture

THE INFERENCE SYSTEM

One of the goals of this project was to maximize the use of existing expert system tools and technology to the extent possible. To this end, several existing expert system techniques and tools were surveyed (Highland, 1985) including OPS5 (Forgy, 1980), EXPERT (Weiss and Kulikowski, 1979), the General Purpose System for Inferencing (GPSI) (Nielsen, 1983, Starbird, 1983), LISP, and PROLOG. The tools were evaluated with respect to their applicability to the problem, ease of use, ability to operate in the workstation environment, expected execution speed, transportability, and other factors. On the basis of this analysis, the University of Illinois GPSI was selected as the basis for further development.

The GPSI system is a backward chaining expert system shell designed to operate on small computer systems. It represents knowledge in the form of decision trees with the goal or conclusion at the top and evidence elements which lead to a conclusion as leaves. To improve execution time performance, the GPSI system features compilation of its knowledge base into an efficient internal form. The system is small, portable, and written entirely in Pascal. The characteristics of small size, speed, and implementation in a compilable, conventional language make the GPSI system a good starting point for embedded systems applications.

Forward chaining reasoning would seem to be the most appropriate for monitoring functions like those required for Shuttle flight control. However, data in the Shuttle environment is highly dynamic and could cause a forward chaining system to perform significant processing in reacting to

rapidly changing data with little inherent significance. In contrast, a properly designed backward chaining system could focus attention on the important aspects of the problem and investigate additional data only when necessary. FLIGHT CONTROLLER uses GPSI's backward chaining reasoning in a cyclic manner with the addition of retained state data. Each set of Shuttle measurements is analyzed for problems and important states stored for later use. In this way processing is minimized and a backward chaining approach is used effectively for a monitoring task.

The data on which FLIGHT CONTROLLER must base its decisions originates in the flight control host computer. Collection and processing of this data is the job of the Information Acquisition and Processing Subsystem (IAPS) of FLIGHT CONTROLLER. Using lists of data names created by the Knowledge Acquisition System (described below) IAPS requests continuous transmission of data values at a regular rate from a central data server in the flight control host computer. On receipt of each set of data values, the data are made available to the inference engine which requests the data through external interface routines as they are needed by the reasoning process.

Other subsystems of the FLIGHT CONTROLLER Inferencing System provide interfaces to workstation executive software and generalized support services. The Display Interface subsystem provides recommendations, explanations, intermediate results, and operating statistics to display services. Display output is the means by which all FLIGHT CONTROLLER recommendations are presented to the user. Directive processing allows the system to accept control directives from the workstation keyboard to start the monitoring process.

A key characteristic of the FLIGHT CONTROLLER Inferencing System design is that it is not a complete system in itself, but an embedded application program within the workstation environment. FLIGHT CONTROLLER, like other applications in the workstation, receives data, processes it, and produces results. These results may be presented to the user through display applications, or may be used by other software to drive additional internal processing. The view of the expert system as a data source allows the details of man machine interface to be separated from the knowledge base and provides more flexibility in the development of user interfaces while giving the user with a consistent interface to all workstation applications.

THE KNOWLEDGE ACQUISITION SYSTEM

Early prototypes using the GPSI system showed that its knowledge representation language had certain undesirable characteristics. Although the decision tree representation of knowledge was easy to understand, the representation language was excessively wordy and difficult to maintain and update. To alleviate this problem and to allow close integration of the knowledge representation with the characteristics of the Shuttle environment, a knowledge representation language preprocessor was developed. The preprocessor, called the Knowledge Acquisition System (KAS), accepts an English-like knowledge representation language which is parsed and converted into rule trees using software created with the YACC parser generating system (Johnson, 1978). The rule trees are translated into the equivalent

GPSI rule text which is then processed by the GPSI rule compiler for use by the Inference System.

The knowledge representation language allows the specification of rules and concepts in an English-like if-then format using keywords to represent commonly used features such as certainty factors and logical relationships. Each rule specifies a set of conditions that lead to an intermediate result or conclusion and can specify one or more actions to be taken. Intermediate results may be used in other rules to simplify the structure of the rule conditions and to encapsulate logical concepts relevant to the problem domain. Multiple rules can be specified for each result or conclusion providing flexibility in the specification of conditions with many possible causes and making rules easier to understand.

The knowledge representation language is designed to allow problem specification in a form pertinent to the flight control environment. Numerical comparison operations on downlinked Shuttle measurements and ground tracking data are the basis for most judgments in this problem domain. These can be specified directly in the knowledge representation language using Shuttle measurement names or their aliases and traditional comparison operators. The use of Shuttle measurement names in the rules results in the automatic creation of tables that allow efficient retrieval of the associated data during real-time operation. Using this approach, the knowledge engineer is freed from defining unique access characteristics for each measurement required. While this may seem like a trivial feature, it is a significant advantage in creating and maintaining knowledge bases involving hundreds of measurements. In a similar fashion, parameters or local variables used during the inferencing process are defined through

their usage in the knowledge base and are made available to the display system. This allows the end user to view intermediate results and current states from the inferencing process for debugging or explanation purposes.

FLIGHT CONTROLLER's reasoning must also address the problem of missing data. Some data items from the Shuttle's telemetry downlink may be unavailable because of degraded equipment modes or weather conditions. To allow for this, FLIGHT CONTROLLER makes use of GPSI's fuzzy logic (Zadeh, 1979) capabilities. As part of the data retrieval process, the status of each Shuttle measurement (supplied by the host data retrieval system) is checked and reflected in a certainty factor associated with the measurement. If the status of a parameter shows that a data value is present, a certainty factor of 1.0 (true) is assigned to that measurement. If the status shows that the measurement's value is not currently available, a certainty of 0.5 (unknown) is assigned. These certainty factors are then propagated to the results of comparison operations and to the conclusions drawn by the rules. This frees the knowledge engineer from being concerned about the availability of telemetry data and simplifies the creation of rules. In addition to the use of fuzzy logic to indicate data availability, FLIGHT CONTROLLER also supports the conventional use of fuzzy logic certainty factors in its rules.

KNOWLEDGE BASE DEVELOPMENT

The approach used to develop the FLIGHT CONTROLLER/ONAV knowledge base took into account the research and prototype nature of this project. The knowledge base was developed using a methodology based on a gradual acquisition of knowledge that more closely follows the human educational process. The highlights of the methodology are:

- Creation of an initial knowledge base using available documentation and training material
- Refinement of the knowledge base through experiences derived from a simulation environment
- Further refinement using domain experts

This approach offers several advantages over the conventional knowledge engineering approach of interviewing experts to extract their expertise. First, the initial knowledge base is based on "first principles" or application theory making the knowledge base more general because it is not based on the specific set of examples that were used in interviewing the domain expert. This also provides a framework to guide the development producing a more complete and better organized knowledge base. Secondly, the use of background and theory information gives the knowledge engineer, as well as the expert system, the proper vocabulary with which to communicate intelligently with domain experts. Third, the domain experts time is utilized more effectively. The domain expert can be used to refine the initial model rather than being required to train the knowledge engineer

and the expert system from the ground up. Finally, if the available documentation is adequate and it is only desired (or possible) to build a demonstration prototype, this approach does not require the use of a domain expert. While the experience of a domain expert is invaluable in the development of an operational system, limited demonstration systems can be built without one.

The documentation used to develop the initial knowledge base was the NASA ONAV Console Procedures manual (NASA, 1982). This document provided a good starting point because it was well written and because the domain experts considered it a good representation of the job they do. In addition, this document contained a number of heuristics based on experience with past NASA spacecraft missions and navigation systems. This information was supplemented with the onboard software requirements document (Rockwell, 1985) which provided information about the design and operation of the onboard software systems.

The use of domain documentation and training material provided a natural organization for the development of the knowledge base. This organization is illustrated by Figure 3. The problem is partitioned along two dimensions - the systems being monitored and the major functions required to perform the monitoring task. This structure provides a complete decomposition of system elements from which rules can be developed for performing the ONAV tasks.

Once coding of the initial knowledge base was complete, testing of the system was begun. The first phase of testing used hand-coded test data to verify each of the major functions the system was to perform. This approach

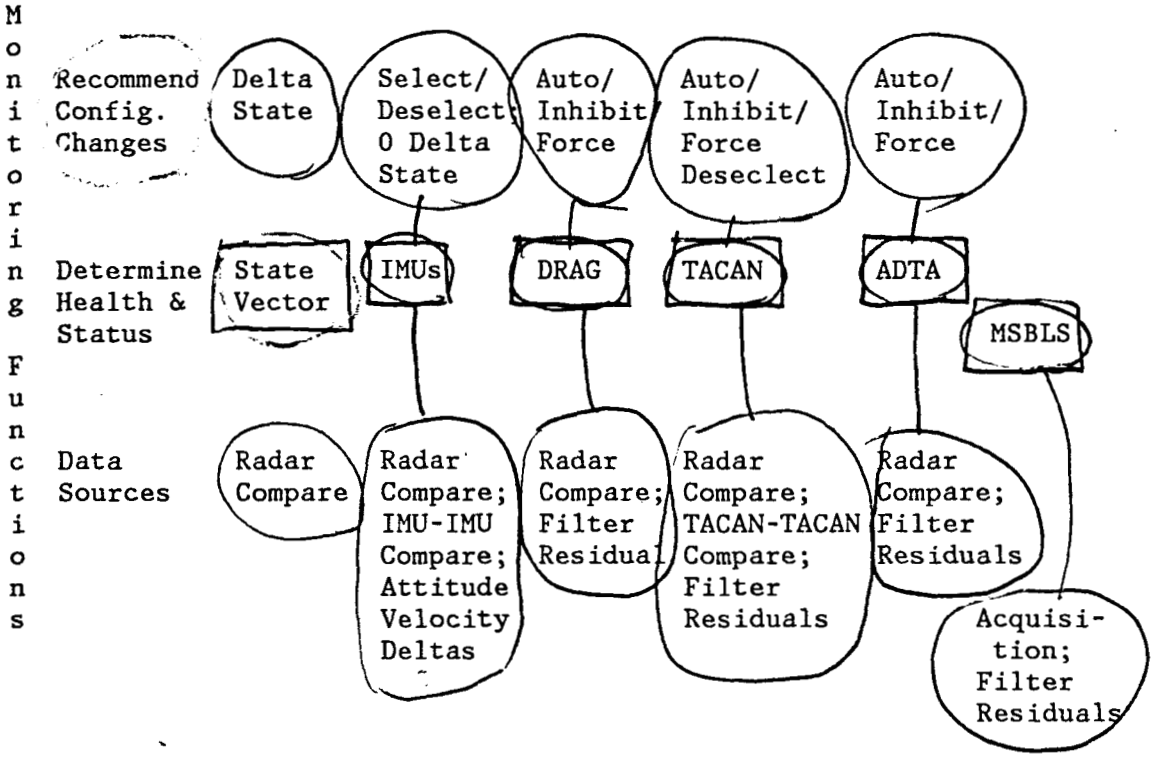


Figure 3. Structural Decomposition of the ONAV Application Knowledge: The decomposition of the ONAV application is based on the navigation system components and the functions required to perform monitoring. This provides a natural organization for the development of the expert system knowledge base.

was useful for initial checkout of the system but was not realistic because the test scenarios were based on the knowledge engineer's expectations of the world and reflected only those rules already coded in the knowledge base. They also lacked the fidelity and complexity of the real environment.

The second phase of testing utilized data from a high fidelity integrated simulation environment. This consisted of an AP 101 Shuttle flight computer loaded with real flight software running in a Flight Equipment Interface Device (FEID) which simulates the Shuttle hardware interfaces. This was driven by avionics and flight dynamics models from the Software Production Facility (SPF) system which is used to test Shuttle flight software. Data from the onboard systems was then captured, transmitted to the Shuttle flight computers containing MCC support software, and then sent over a Local Area Network to a workstation containing FLIGHT CONTROLLER/ONAV. This configuration provides an environment in which high fidelity scenarios can be easily created and used to test the expert system in a closed-loop test environment. This allows expert system recommendations to be immediately implemented through simulated crew actions to effect the overall state of the environment. This use of a high fidelity simulation environment gave the FLIGHT CONTROLLER/ONAV system a large experience base that was used to correct and enhance its knowledge.

RESULTS.

Several scenarios were developed to refine and test the capabilities of the FLIGHT CONTROLLER/ONAV system. A description of one of the more interesting scenarios is given in Figure 4. This scenario involves a nominal Shuttle entry followed by a sequence of navigation component failures that requires the expert system to make time critical decisions. Other scenarios that were developed test the expert system with nominal entries, failures in each major class of devices and systems, and monitoring without the use of ground radar data. These scenarios represent a wide range of problems typical of those that are seen in operational or simulated situations.

The FLIGHT CONTROLLER system software has been implemented and tested on the MASSCOMP workstation with a knowledge base for Primary Flight Control System (PFS) Shuttle landing phase monitoring. The current knowledge base contains 393 rules, uses 140 Shuttle measurements as a data source, and contains an additional 37 internal parameters. It required approximately a manyear to develop the FLIGHT CONTROLLER system software and the ONAV knowledge base.

The system has been run with various scenarios in the integrated flight control environment. It performs an inferencing cycle (producing a set of recommendations on a single set of 140 input measurements) in 4.0 seconds of elapsed time using 4.0 seconds of CPU. This level of performance is adequate to keep up with the real data, which updates every 4.0 seconds. With the advent of more powerful workstations it will be possible to attain even better performance.

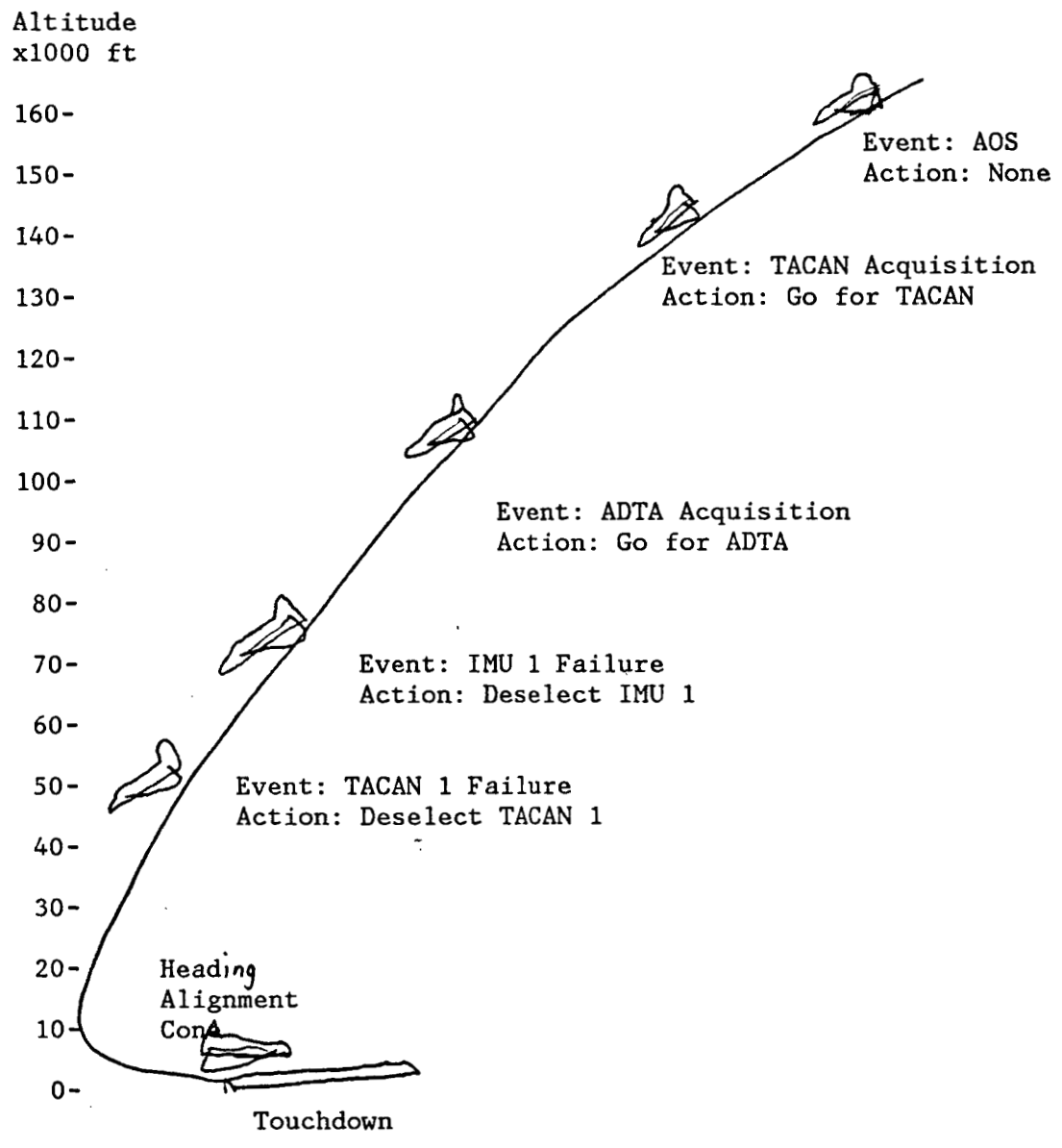
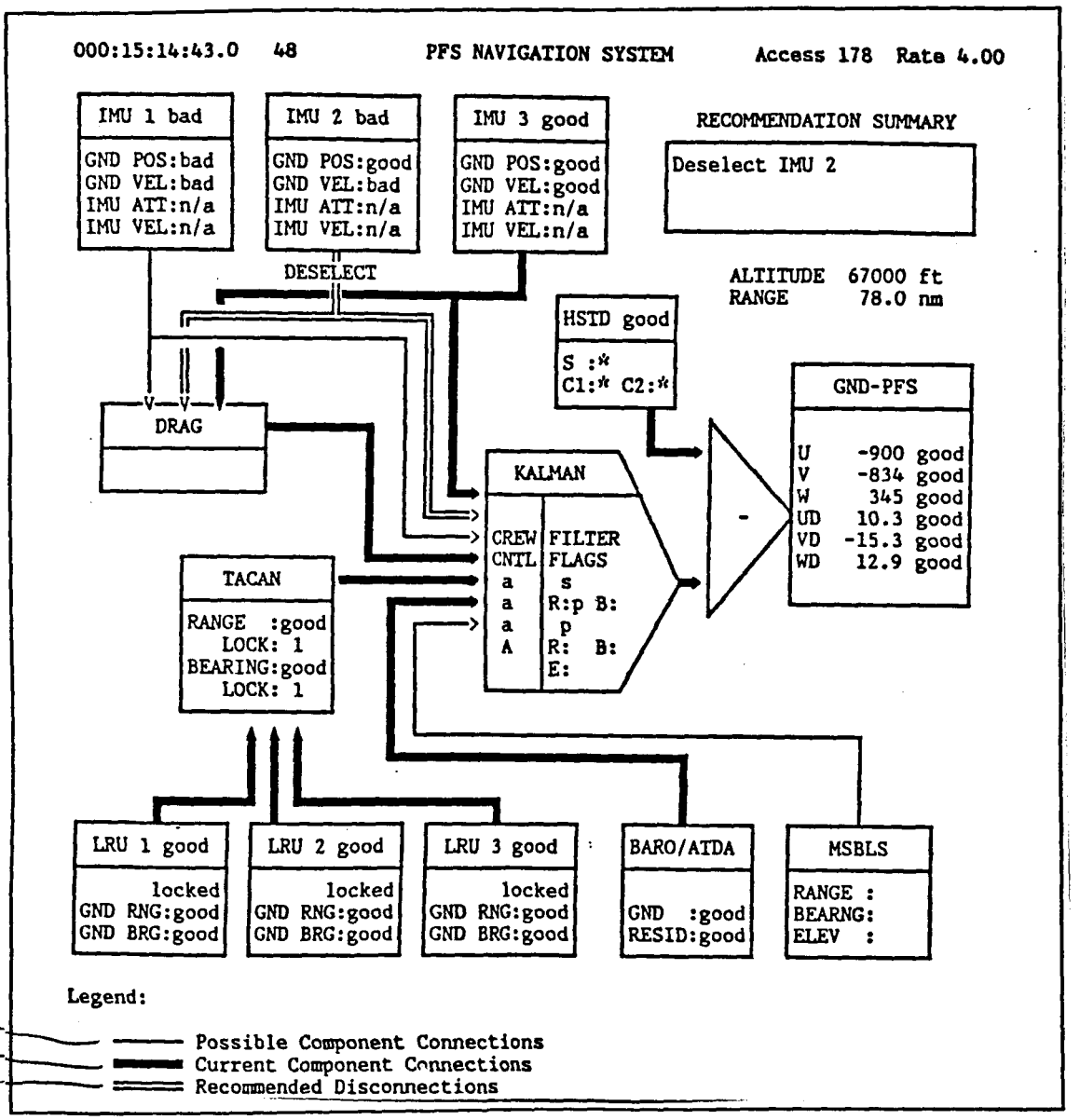


Figure 4. A Typical Shuttle Entry Test Scenario: In this scenario, the expert system must make nominal recommendations to use the TACAN and ADTA systems, then must analyze critical problems in the IMUs and TACAN units to ensure a save Shuttle landing.

The recommendations produced by FLIGHT CONTROLLER were presented on a graphical display like that of Figure 5. The boxes along the outer top, left, and bottom edges of the display represent the components that provide information to the navigation system: the three IMUs, the drag processor, the three TACAN sensors, the Air Data system (ADTA), and the Microwave Landing System (MSBLS). Each of these components is connected to a Kalman filter which combines the sensor inputs to produce a composite onboard position and velocity estimate. This estimate then compared with the ground radar position and velocity (indicated by the box in the upper right) to produce an estimate of the error in the onboard navigation system. The display makes extensive use of color to indicate the current and recommended use of the components. The color scheme is as follows: green indicates that the component is being used, black indicates it is not, yellow indicates that the component should be connected (selected), red indicates that the component should be disconnected (deselected). The use of color allows the systems recommendations to be immediately evident without reading text messages. Short text messages are also provided in the recommendation summary box in the upper left corner of the screen.

Each of the component boxes on this display also contains a detailed status of the characteristics that determine its health indicated by GOOD, BAD, blank, or ??? indicators. This representation of health provides a means of explaining the actions recommended in a manner that can be quickly interpreted by the user and does not impose a significant computation cost since the information is generated as a side effect of the monitoring process.



Black

Green

Red

Figure 5. The FLIGHT CONTROLLER PFS Navigation System Display: This display represents a schematic diagram of the Shuttle Onboard Navigation system with expert system recommendations shown through the use of graphics and color.

CONCLUSIONS.

The work on the FLIGHT CONTROLLER/ONAV system has resulted in a tool for the development of highly integrated and embedded real-time expert systems for the Shuttle flight control environment and has shown the potential of expert systems to automate the flight control tasks. While the current level of expertise in FLIGHT CONTROLLER/ONAV is less than that of experienced ONAV controllers, it appears that the system, with planned improvements, could perform the monitoring task as well as human console operators. Additionally, it has been proven that a real-time monitoring task can be performed using conventional hardware and software and that special purpose programming languages, such as LISP, and special purpose hardware are not necessary.

The knowledge representation language used by FLIGHT CONTROLLER has shown that the knowledge bases of spacecraft control systems should be integrated with real-time data retrieval and presentation capabilities of the system in a transparent manner. This frees the knowledge engineer from writing additional rules and logic to acquire information, allows knowledge bases to be independent of the peculiarities and protocols of data retrieval systems, and makes modification and enhancement of the knowledge base easier. Additionally, information derived by the reasoning process should also be integrated with the surrounding display and data retrieval systems to simplify information presentation and distribution and free the knowledge engineer from issues of man machine interface.

Finally, this project has shown that the availability of a realistic test environment is essential to the development of operational quality expert systems. Expert systems are intended to operate in complex environments which can not be simulated realistically using simple test drivers. A high fidelity simulation environment or test cases drawn from real world situations is essential for the development of a system that is expected to produce expert level performance.

The FLIGHT CONTROLLER/ONAV system has been demonstrated to USAF and NASA personnel using a complex off-nominal Shuttle entry scenario in which the expert system must play the role of an ONAV flight controller in recommending the proper crew actions to save the Shuttle from a series of navigation equipment failures. NASA ONAV console personnel have been impressed with the systems high level of integration, real-time capabilities, and realistic interpretation of the Shuttle data. NASA is currently funding research to continue the development of the ONAV knowledge base and to install the FLIGHT CONTROLLER system in the NASA development laboratories. After additional testing, upgrading, and evaluation, the FLIGHT CONTROLLER/ONAV system may be used to assist ONAV console operators during Shuttle training simulations and missions.

REFERENCES.

Forgy, C. L., 1980. The OPS5 User's Manual. Technical Report, Carnegie-Mellon University.

Highland, F. D., 1985, Design of an Expert System for Shuttle Ground Control. Masters Thesis, University of Houston Clear Lake.

Johnson, S. C., 1978. YACC: Yet Another Compiler-Compiler. Bell Labs, Murry Hill, N.J.

NASA, 1982. Onboard Navigation Console Procedures. NASA 81-FM-23 JSC-17311.

NASA, 1985. Advancing Automation and Robotics Technology for the Space Station and for the U.S. Economy. NASA Technical Memo 87566 Vol 1, March 1985.

Nielsen, P., 1983. A User's Manual for Construct and Consult in the GPSI Environment. University of Illinois KBPA Project.

Rockwell International, 1985. Space Shuttle Orbital Flight Test Level C Functional Subsystem Software Requirements Document, Guidance, Navigation, and Control: Part B, Entry through Landing Navigation. Document STS 83-0004A.

Starbird, R. P., and Ashford, T. J., 1983. Using an Expert System to Diagnose Hardware: An Application of GPSI. in Proceedings of the IBM Expert Systems - ITL Conference, pp. 108-147.

Waterman, D., 1985. A Guide to Expert Systems. Addison Wesley, Reading, Mass.

Weiss, D., and Kulikowski, C. A., 1979. EXPERT: A System for Developing Consultation Models. Proceedings of IJCAI-79, pp. 942-947.

Zadeh, L. A. 1979. Approximate reasoning based on fuzzy Logic. Proceedings of IJCAI-79. pp. 1004-1010.